Class 5: Personal Data Projects,Visualization Forms, Plus Arrays

Fall2012/New School
zannah marsh
zmarsh10@gmail.com

Plan

- A few personal data projects, plus lecture on pies, bars, lines, stacks and streams
- Break
- Processing/Arrays
- Stephanie and Jessica present

Data Projects:

- Personal Documentation
- Sonification (if time)

Graphic Design Principles

- Contrast
- Proximity
- Alignment
- Repetition

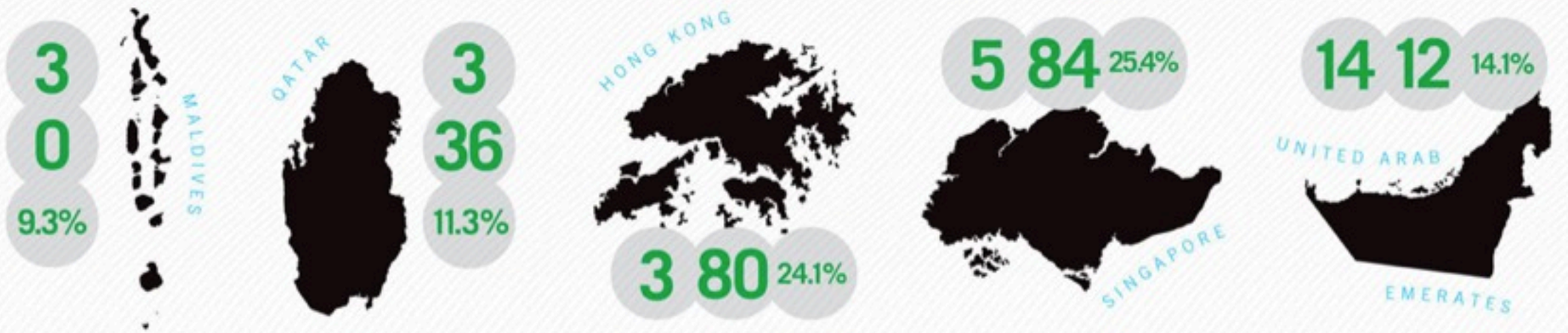
Taxes Around The World

You probably just received your tax forms in the mail. If you think your taxes are complicated, though, imagine a business's tax forms. PricewaterhouseCoopers recently completed a survey of how easy it is for businesses to pay taxes in every country in the world, based on a variety of categories. The United States ranked 64th out of 183 countries surveyed. Here are, for comparison, the countries with the simplest (and cheapest) and most complicated (and most expensive) tax systems.

A B C

A Number of payments a year
B Hours per year spent complying
C Total tax rate (sum of profit, labor, and consumption taxes)

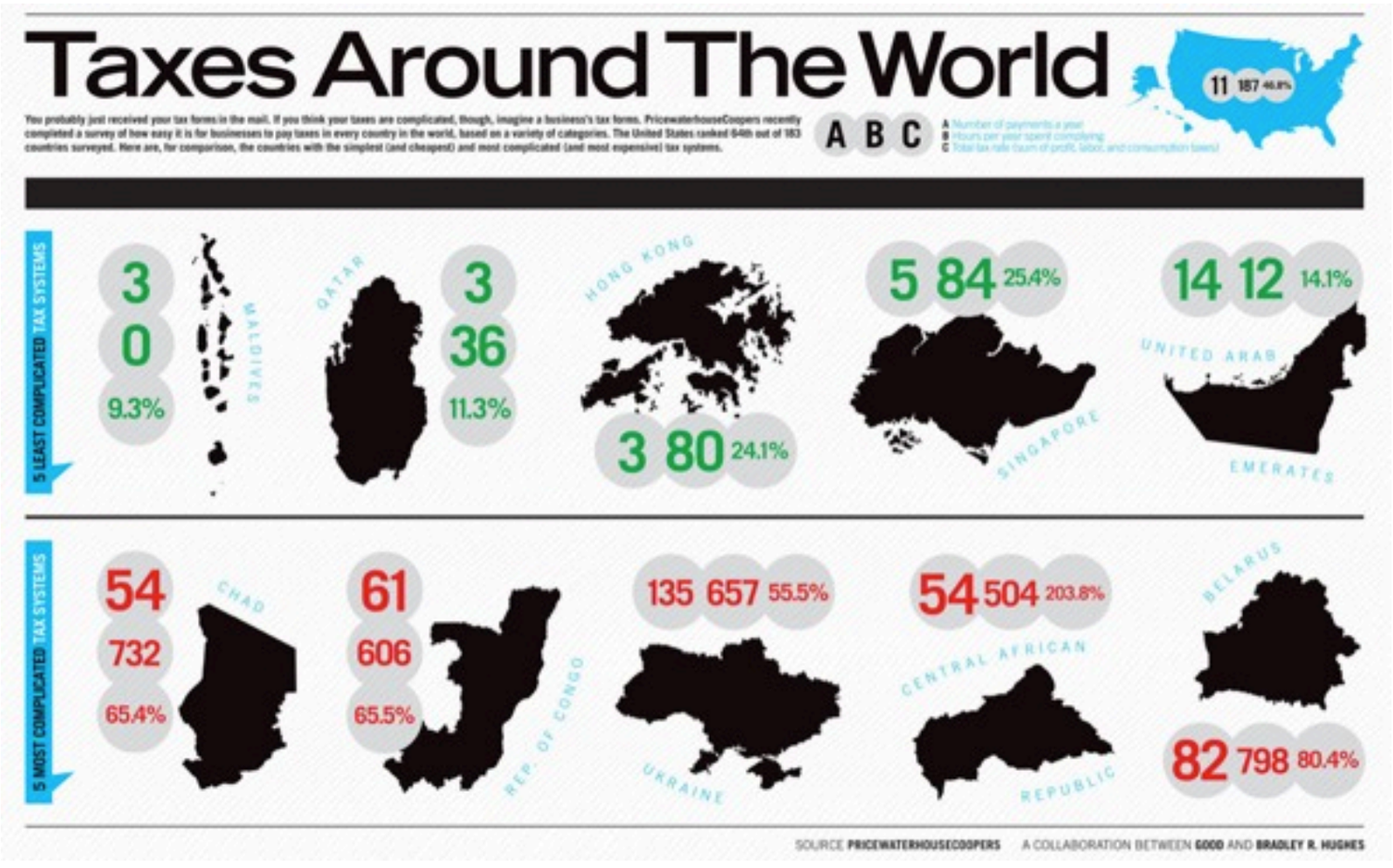
5 LEAST COMPLICATED TAX SYSTEMS



5 MOST COMPLICATED TAX SYSTEMS

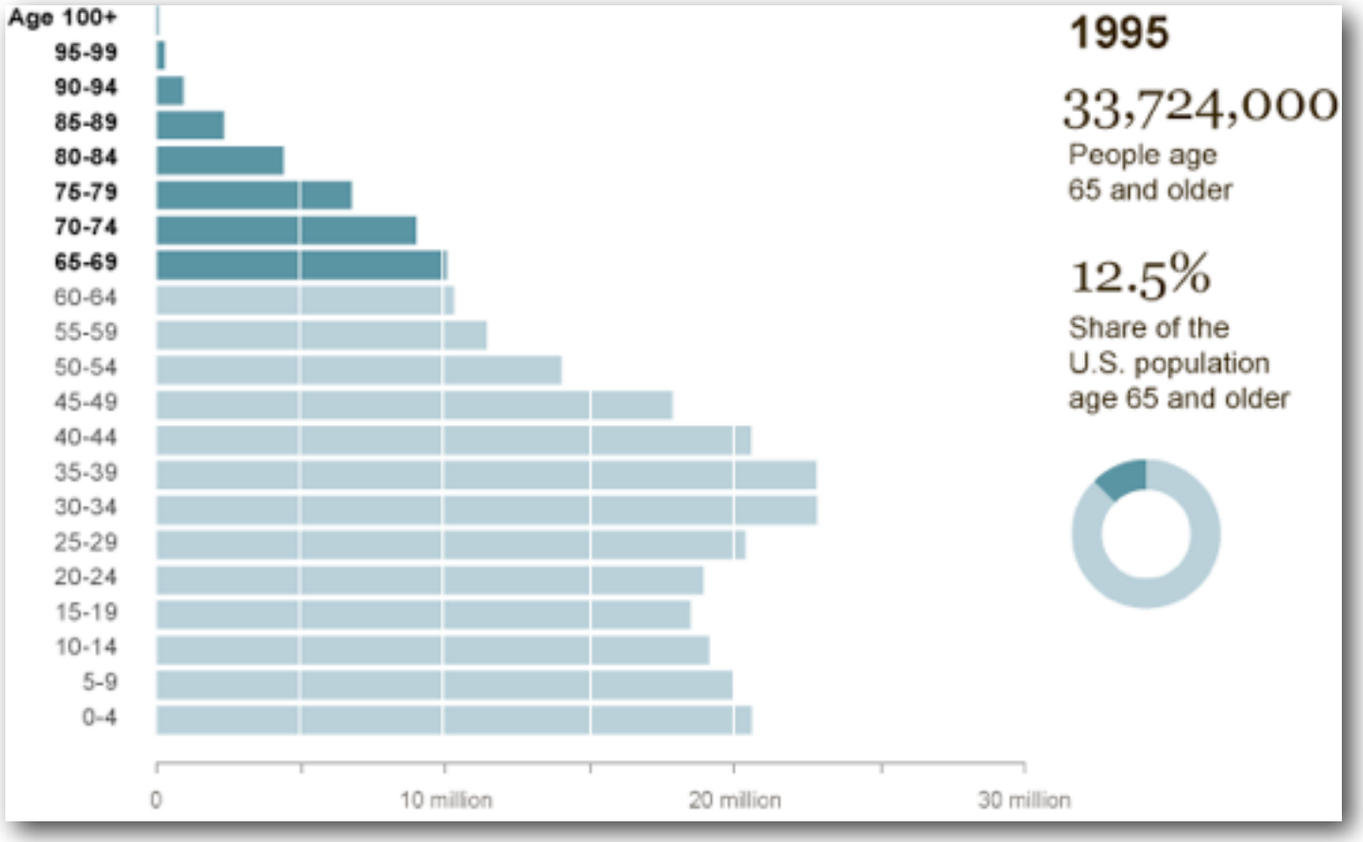
SOURCE **PRICEWATERHOUSECOOPERS** A COLLABORATION BETWEEN **GOOD** AND **BRADLEY R. HUGHES**

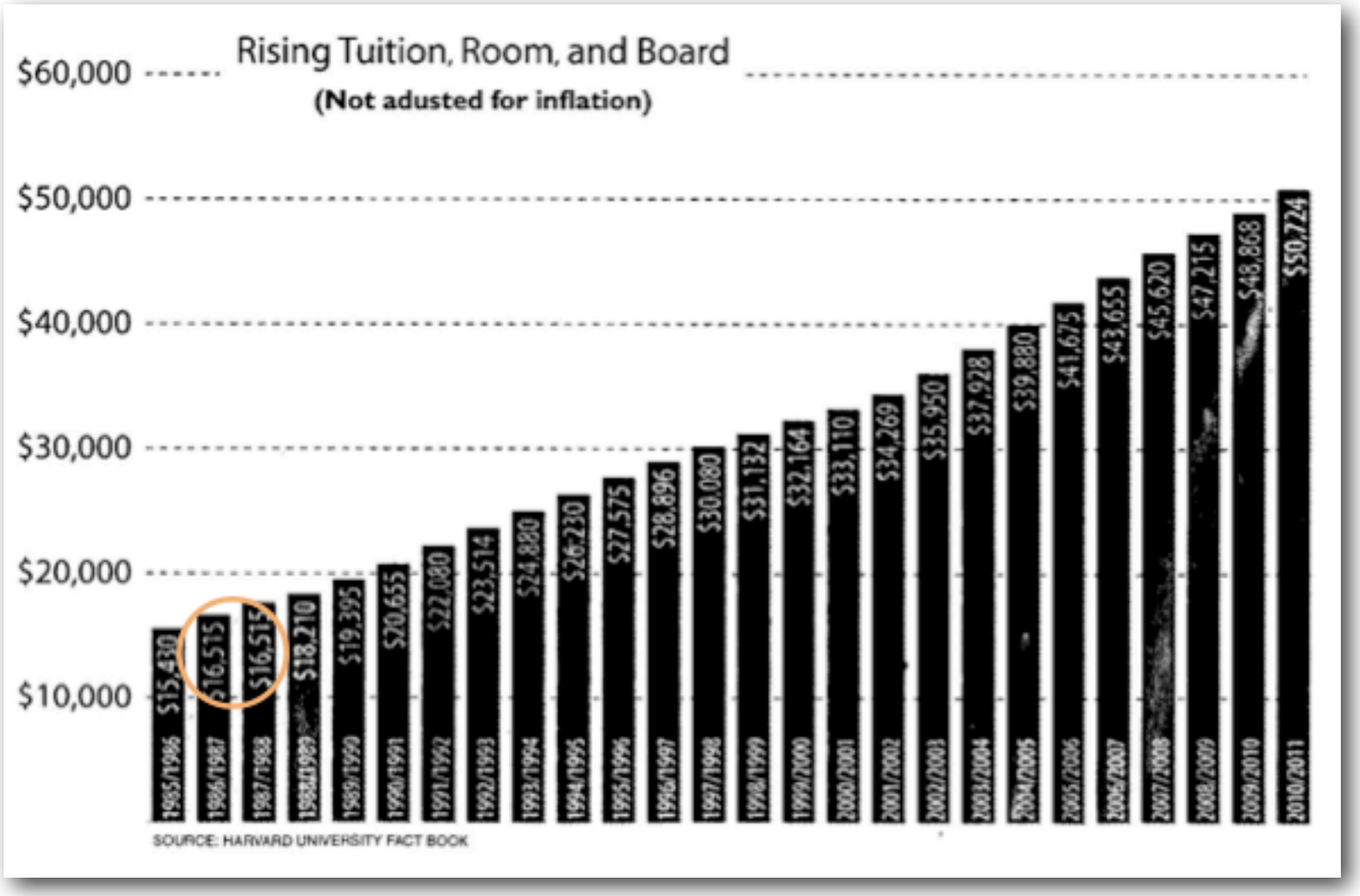
Graphic Vs. Table?

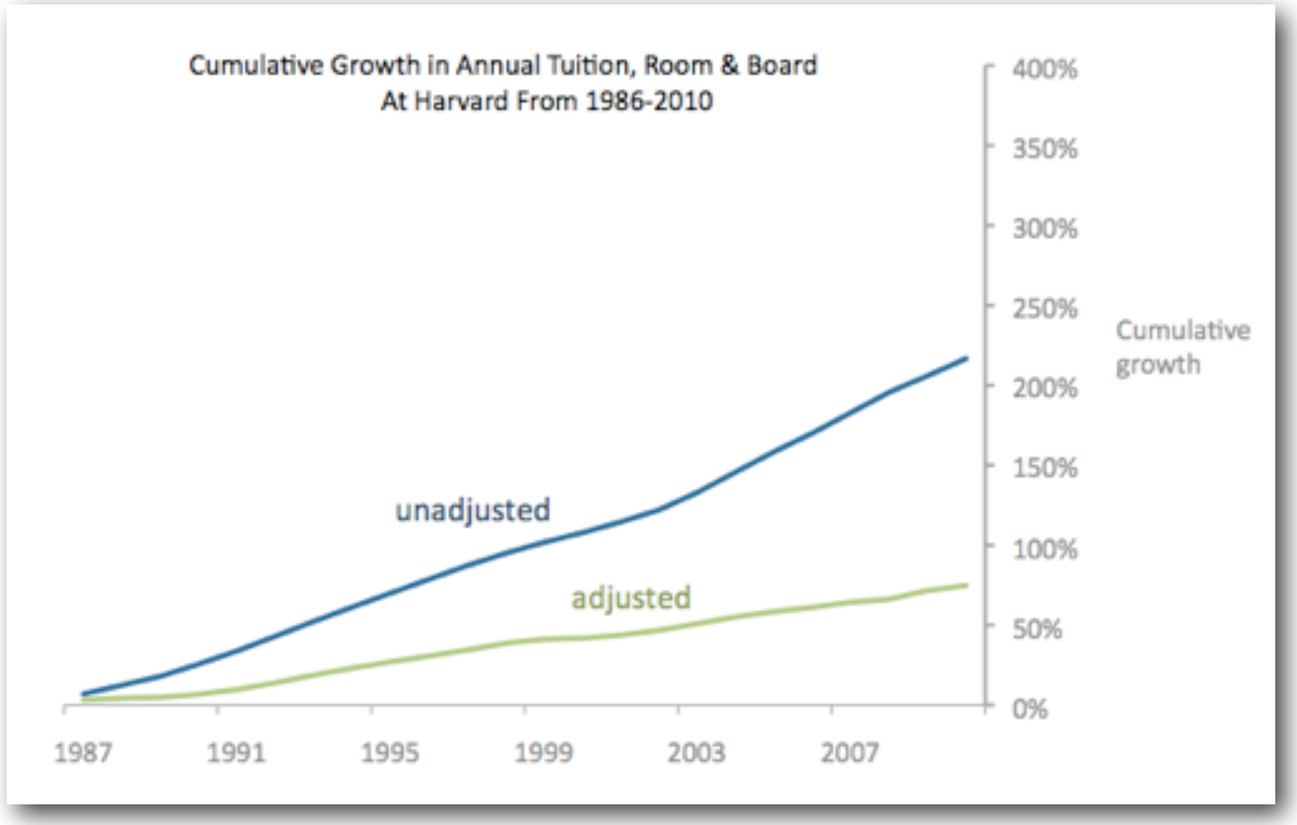


country	# of payments/year	hours spent on prep	total tax rate (%)
Maldives	3	0	9.3
Quatar	3	36	11.3
Hong Kong	3	80	24.1
Singapore	5	84	25.4
UAE	14	12	14.1
Chad	54	732	65.4
Rep of Congo	61	606	65.5
Ukraine	135	657	55.5
Central African Republic	54	505	203.8
Belarus	83	798	80.4

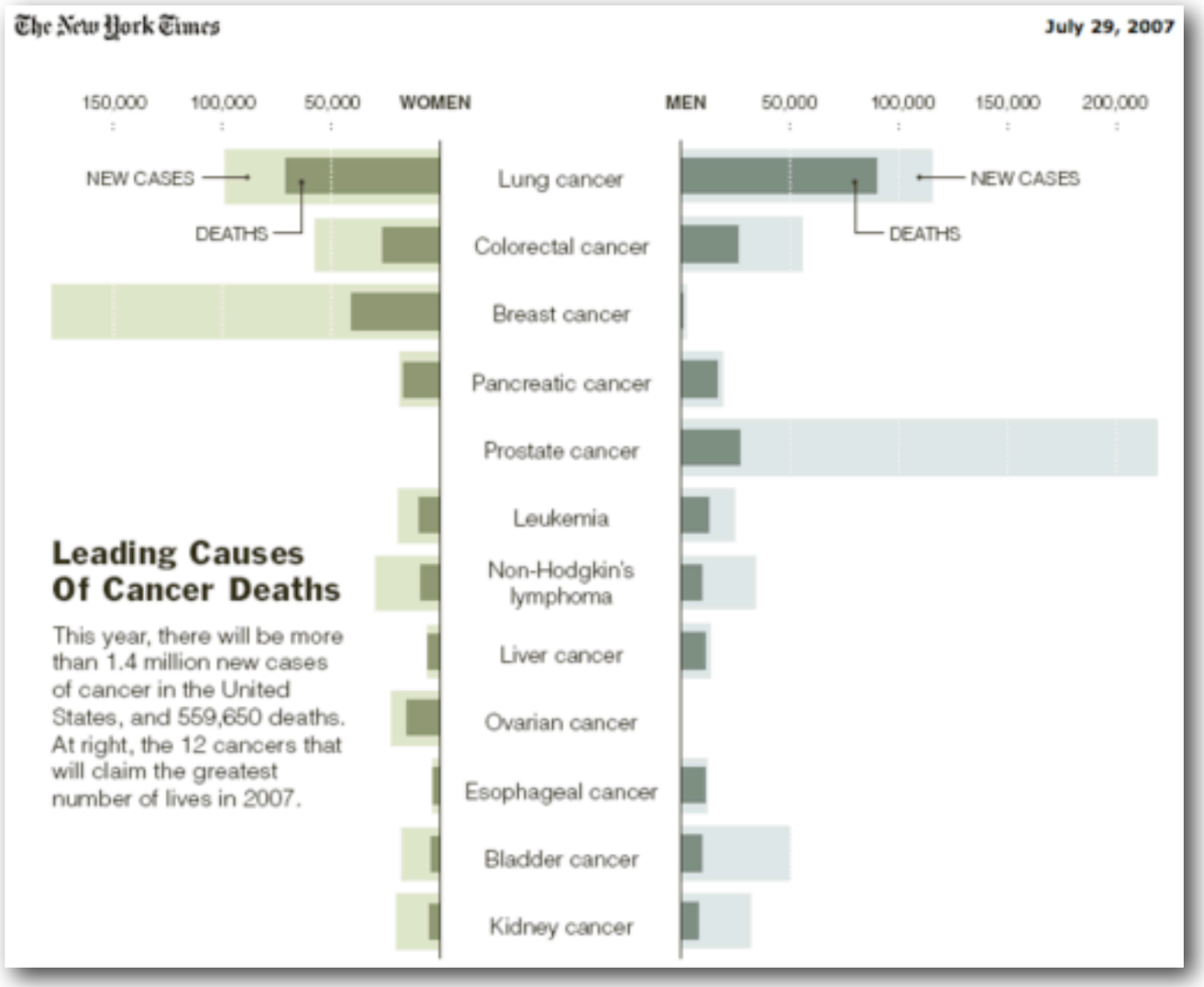
Bar Chart

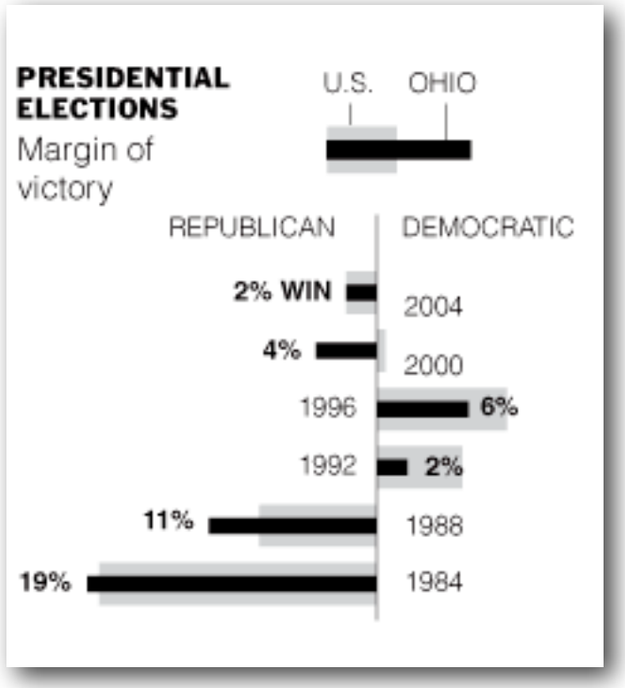

Age 100+
95-99
90-94
85-89
80-84
75-79
70-74
65-69
60-64
55-59
50-54
45-49
40-44
35-39
30-34
25-29
20-24
15-19
10-14
5-9
0-4
0
10 million
20 million
30 million
1995
33,724,000
People age
65 and older
12.5%
Share of the
U.S. population
age 65 and older

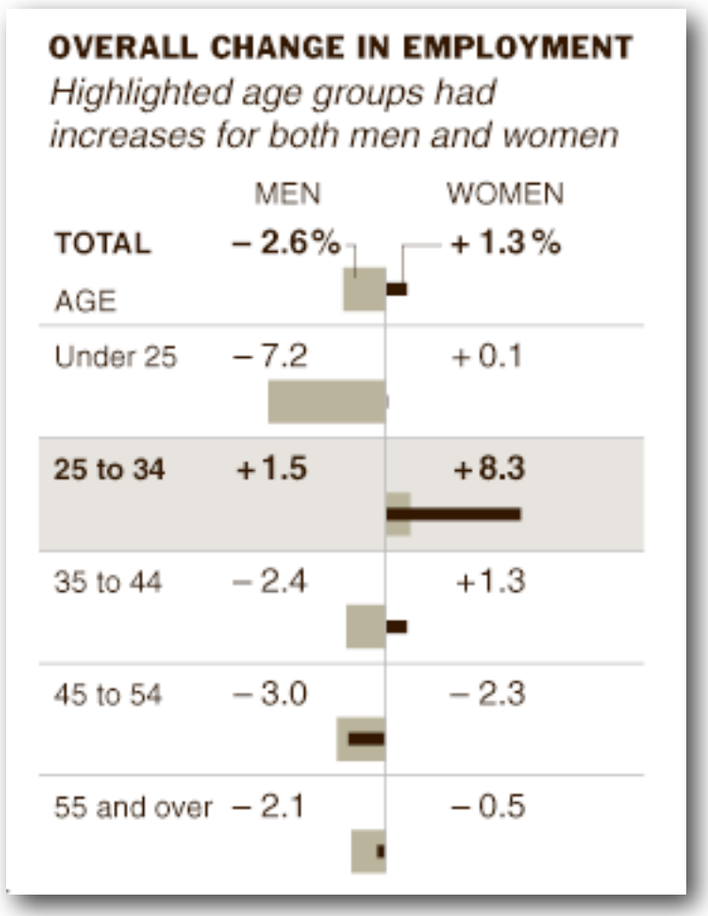

Rising Tuition, Room, and Board
(Not adusted for inflation)
$60,000
$50,000
$40,000
$30,000
$20,000
$10,000
$15,430
$16,515
$16,515
$18,210
$19,395
$20,655
$22,080
$23,514
$24,880
$26,230
$27,575
$28,896
$30,080
$31,132
$32,164
$33,110
$34,269
$35,950
$37,928
$39,880
$41,675
$43,655
$45,620
$47,215
$48,868
$50,724
1986/1987
1987/1988
1988/1989
1989/1990
1990/1991
1991/1992
1992/1993
1993/1994
1994/1995
1995/1996
1996/1997
1997/1998
1998/1999
1999/2000
2000/2001
2001/2002
2002/2003
2003/2004
2004/2005
2005/2006
2006/2007
2007/2008
2008/2009
2009/2010
2010/2011
SOURCE: HARVARD UNIVERSITY FACT BOOK


Cumulative Growth in Annual Tuition, Room & Board
At Harvard From 1986-2010
400%
350%
300%
250%
200%
150%
100%
50%
0%
Cumulative growth
unadjusted
adjusted
1987
1991
1995
1999
2003
2007

bar chart- more complex

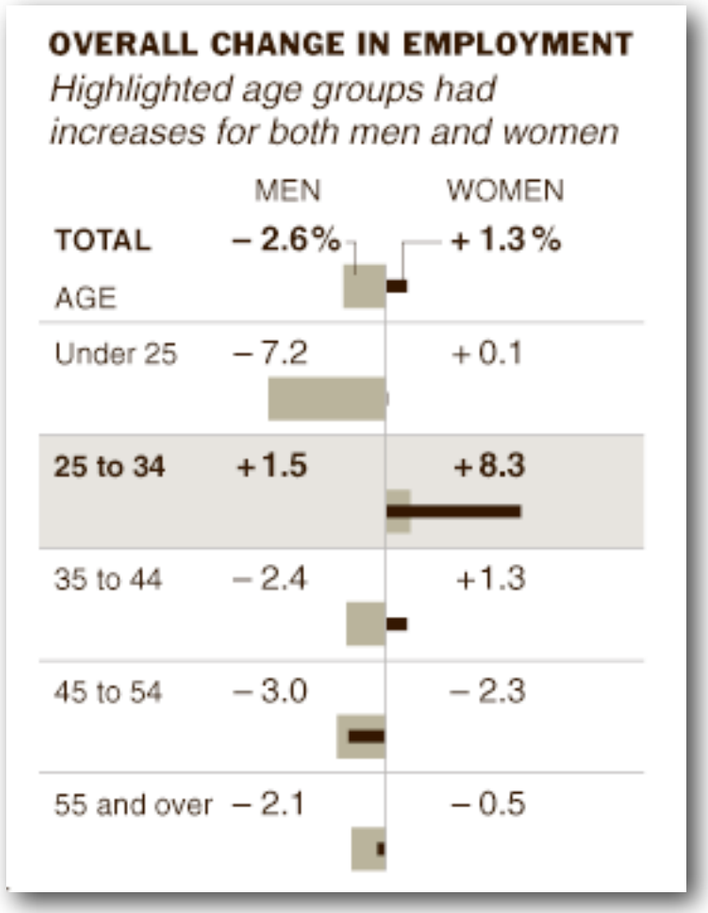


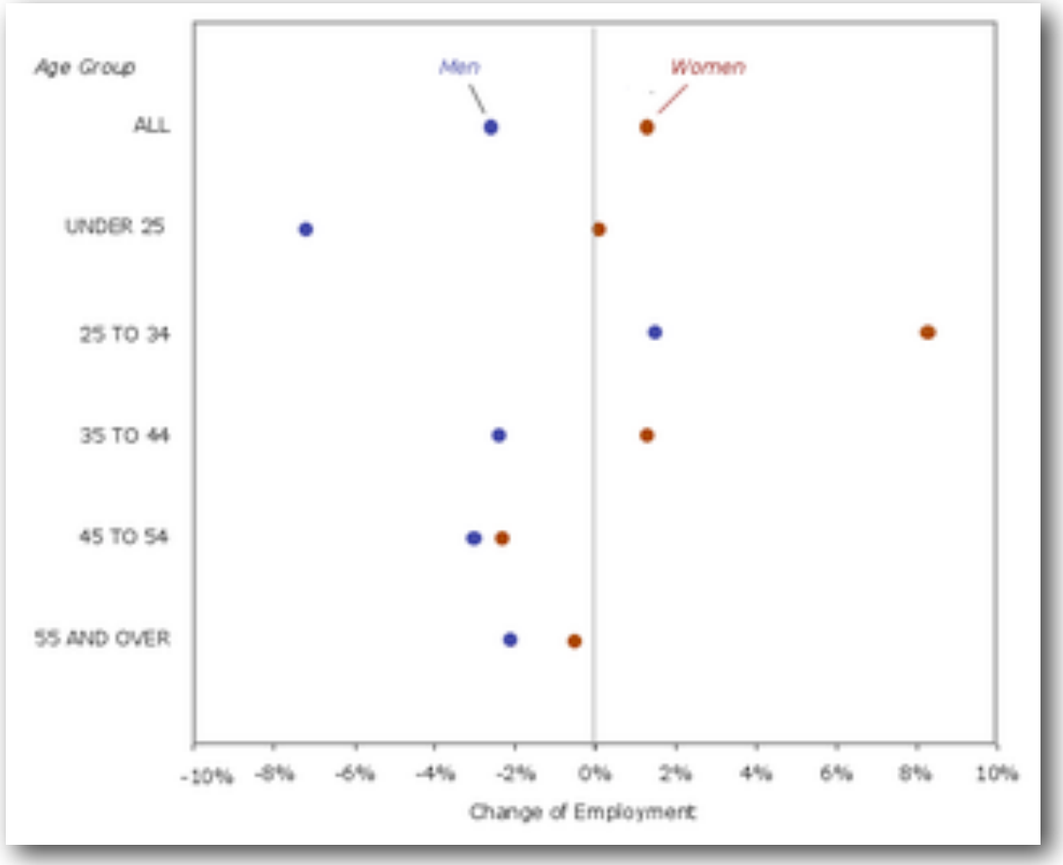




bar chart-table hybrid

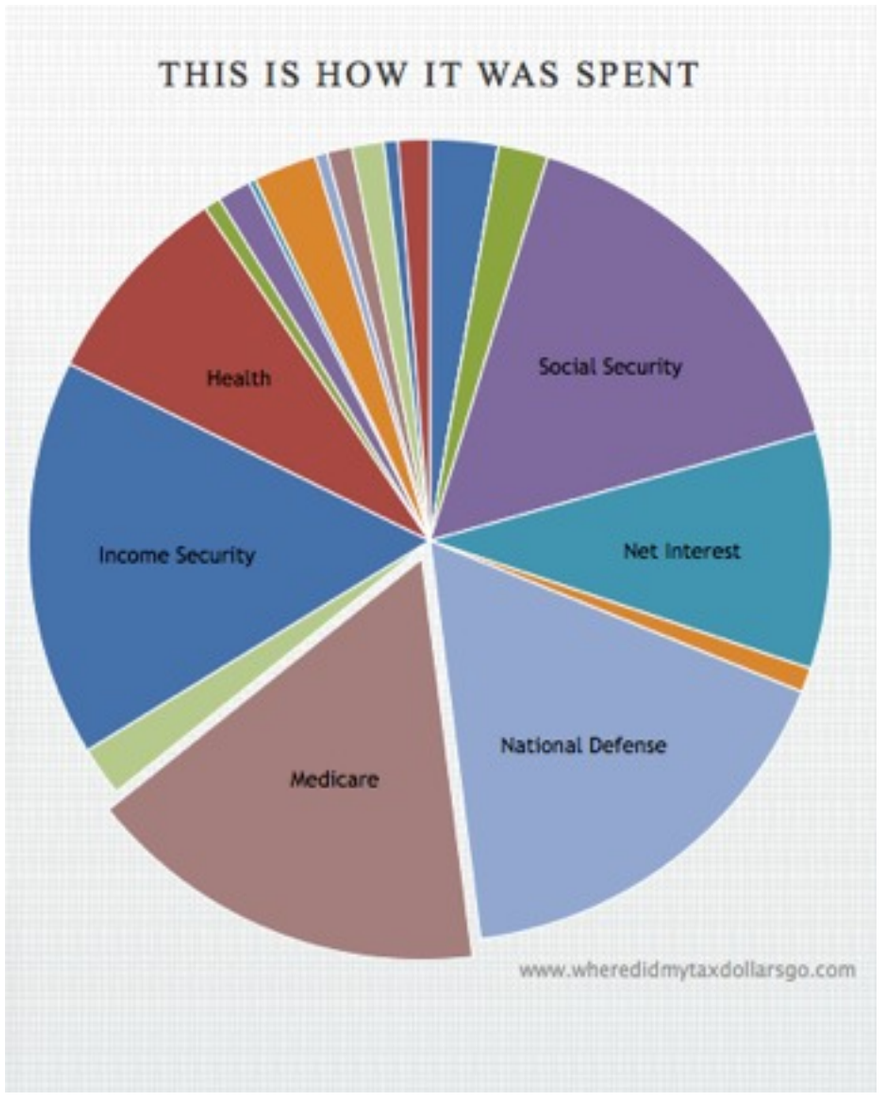
Age	Men	Women
ALL	-2.6%	1.3%
UNDER 25	-7.2%	0.1%
25 TO 34	1.5%	8.3%
35 TO 44	-2.4%	1.3%
45 TO 54	-3.0%	-2.3%
OVER 55	-2.1%	-0.5%

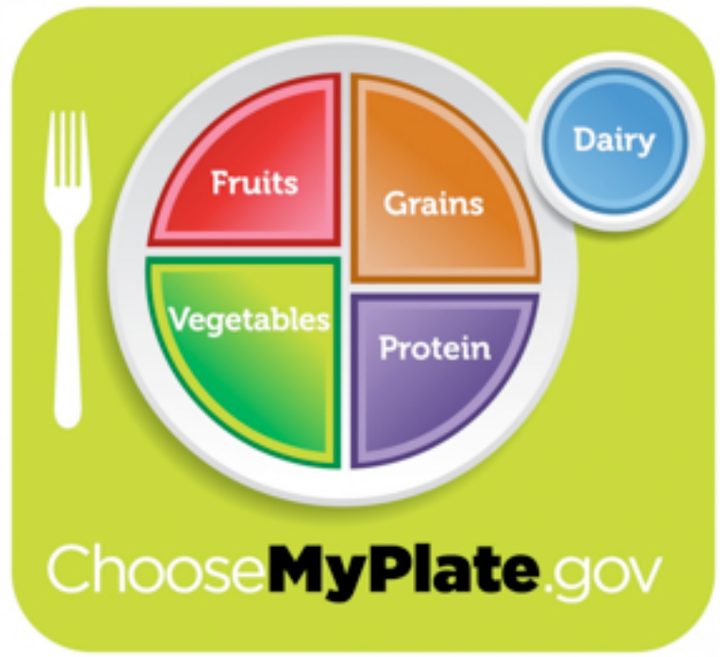


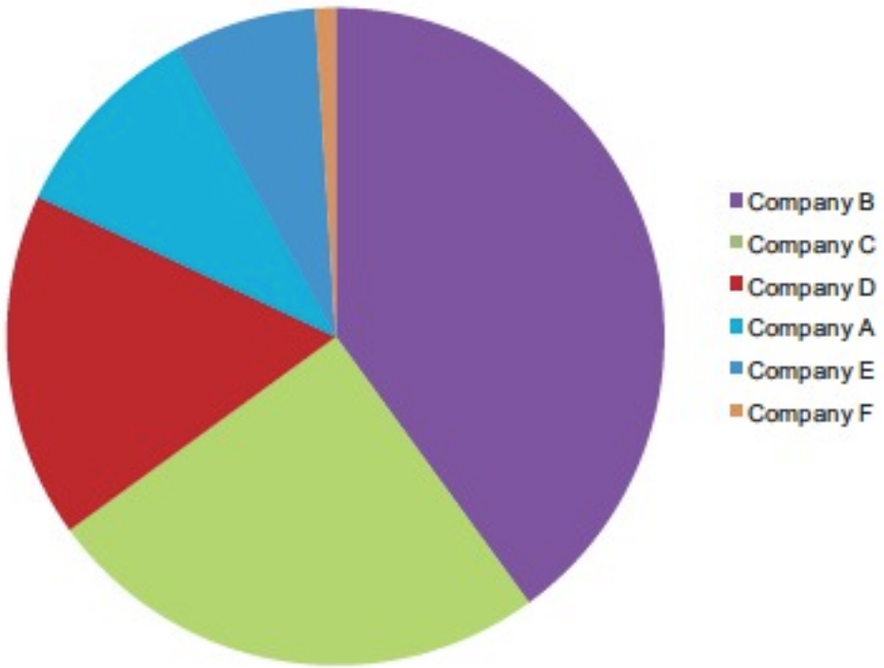


http://junkcharts.typepad.com/junk_charts/2009/11/disentangle.html

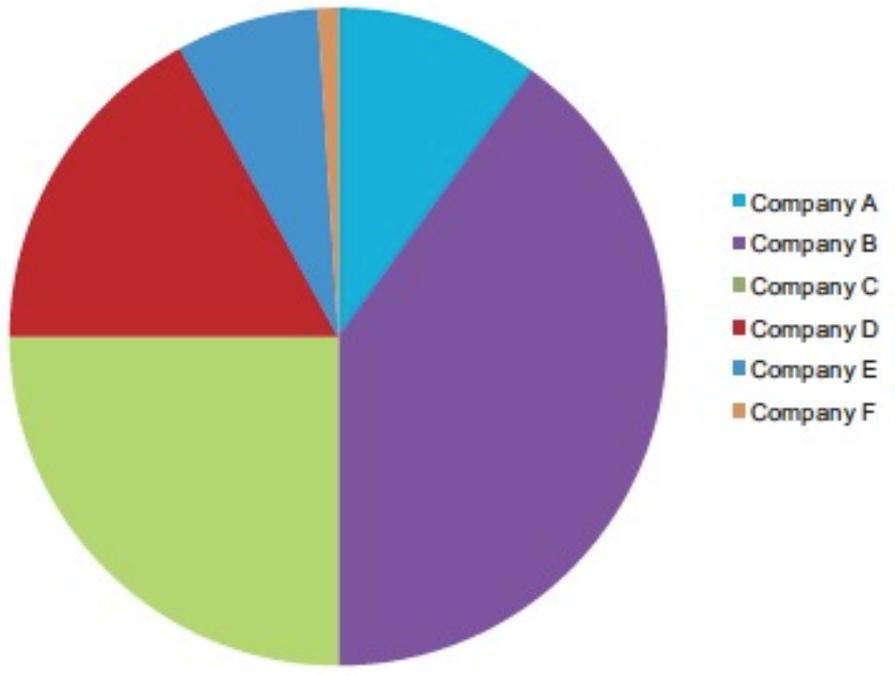
Pies and Donuts

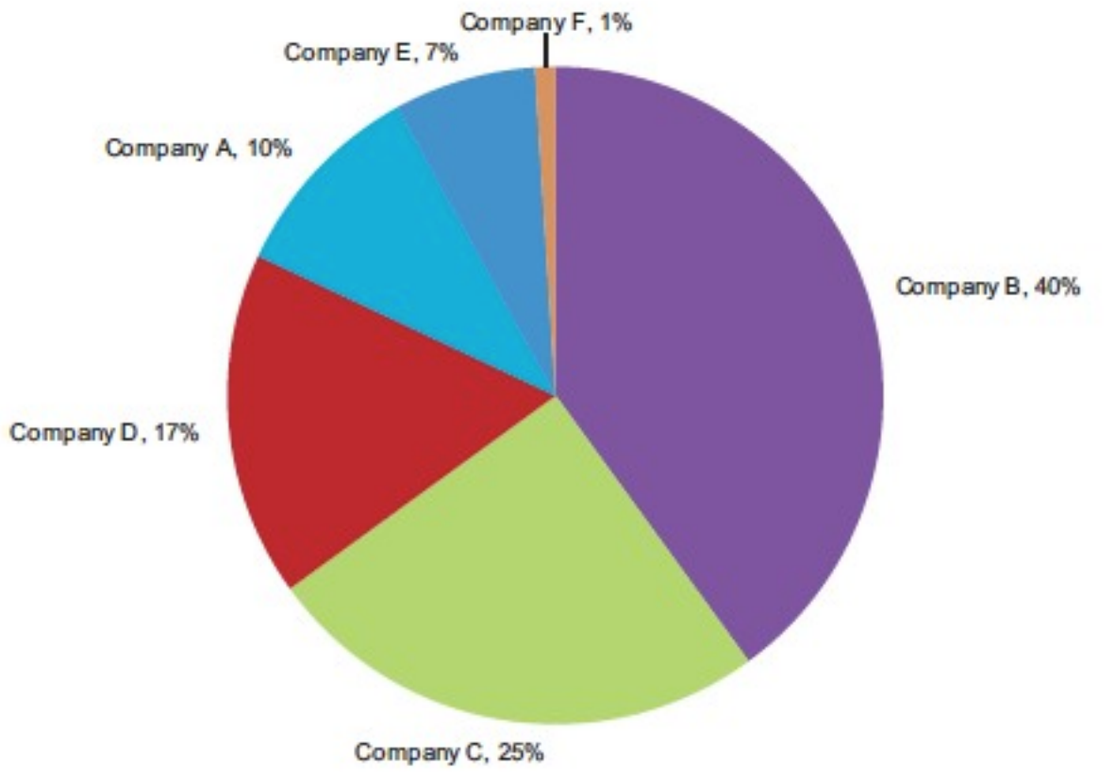


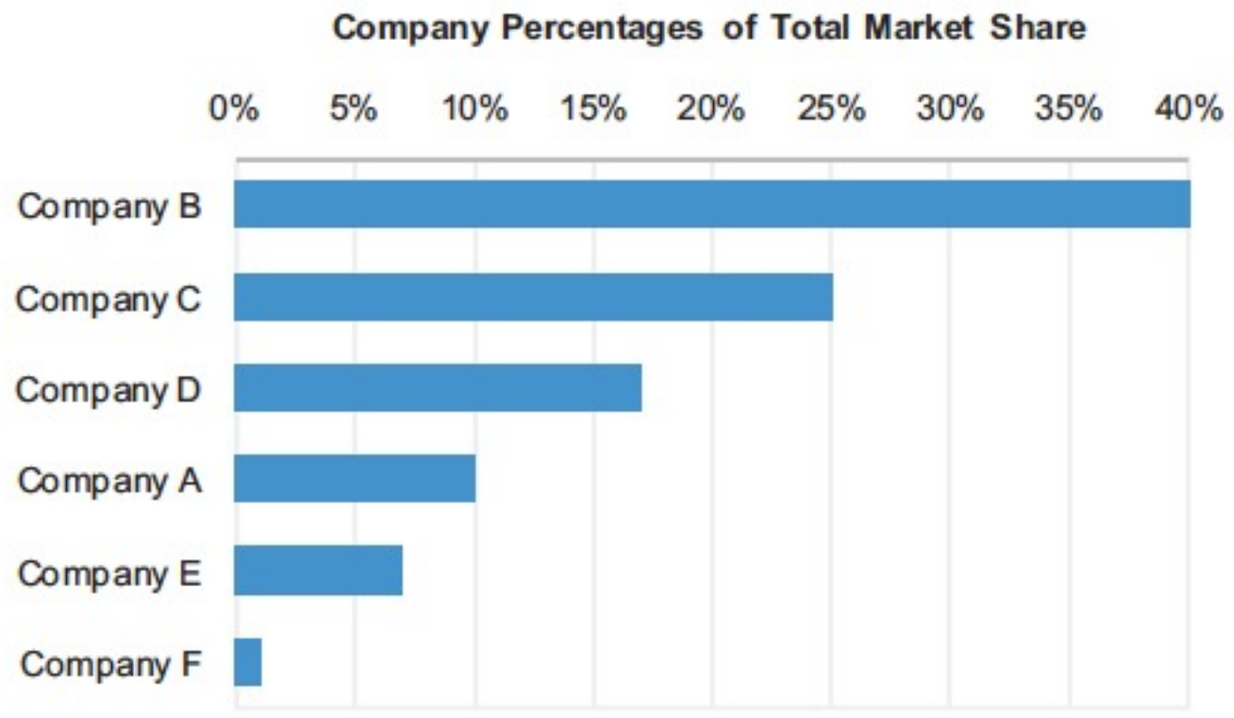




what percent of the whole?

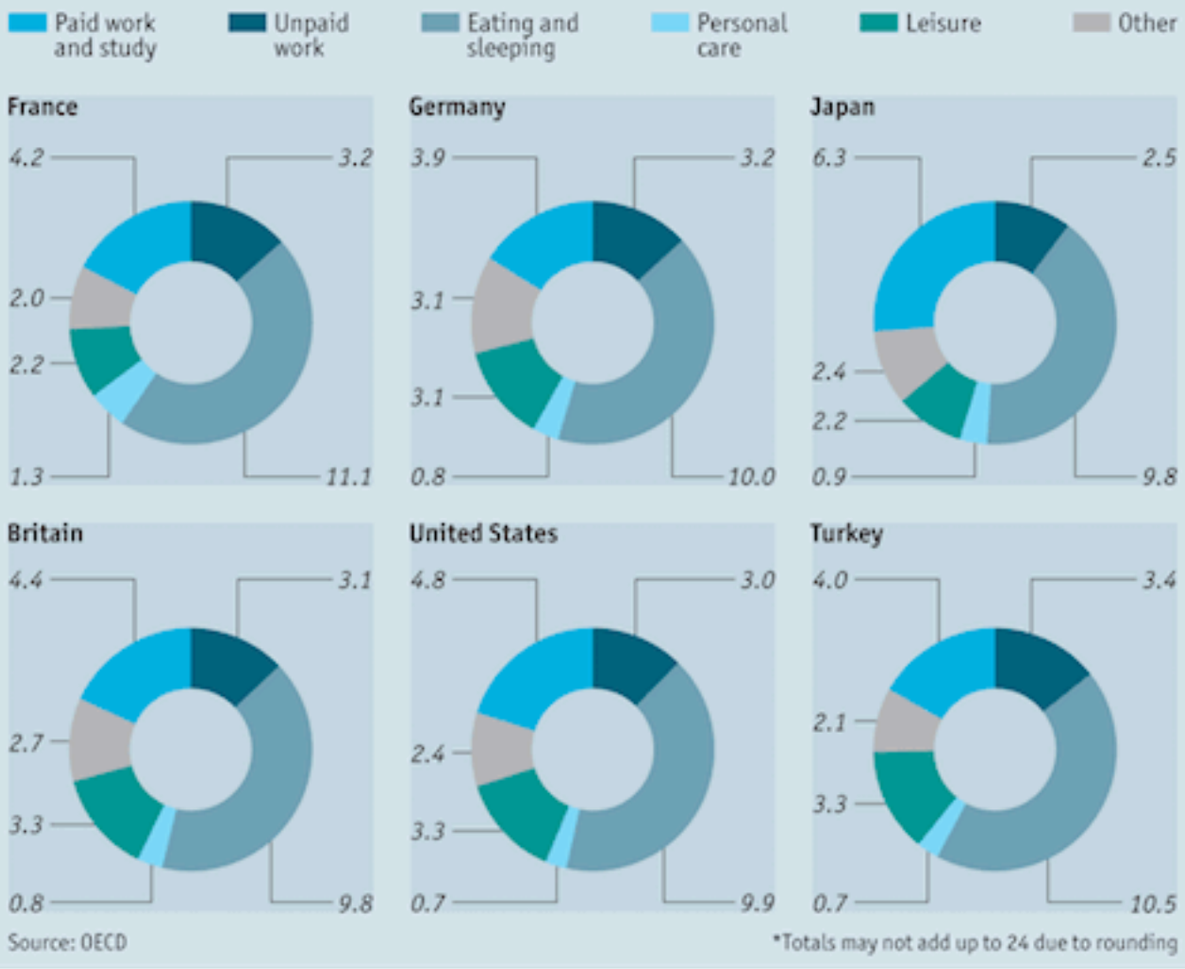



Company F, 1%
Company E, 7%
Company A, 10%
Company B, 40%
Company D, 17%
Company C, 25%


Company Percentages of Total Market Share
0%
5%
10%
15%
20%
25%
30%
35%
40%
Company B
Company C
Company D
Company A
Company E
Company F

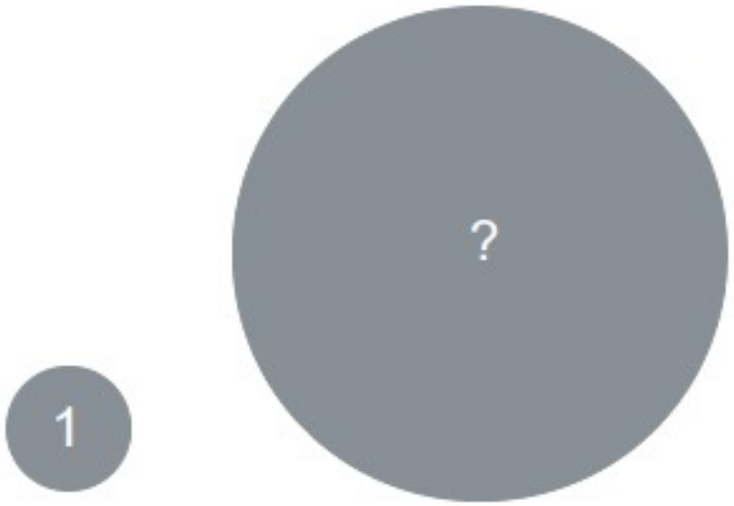
Time spent per activity

Hours per day by people aged 15-64, 1998-2009

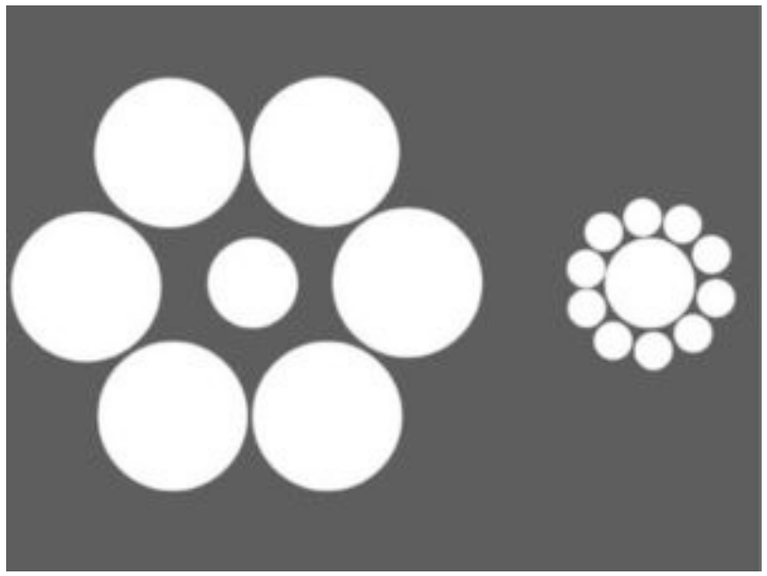


Source: OECD

*Totals may not add up to 24 due to rounding

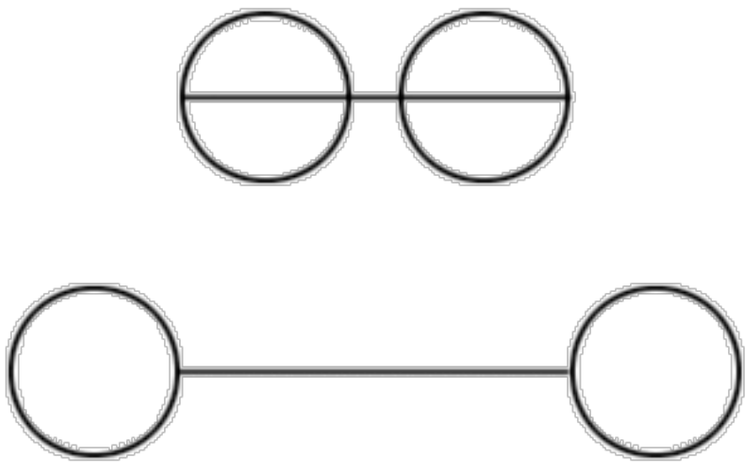


Can we judge relative areas of circles?

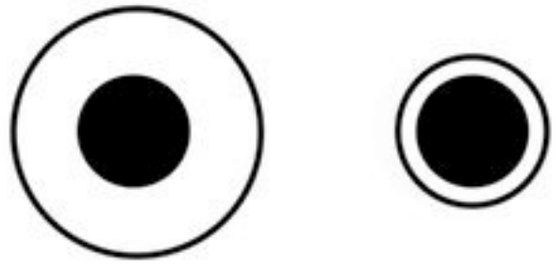


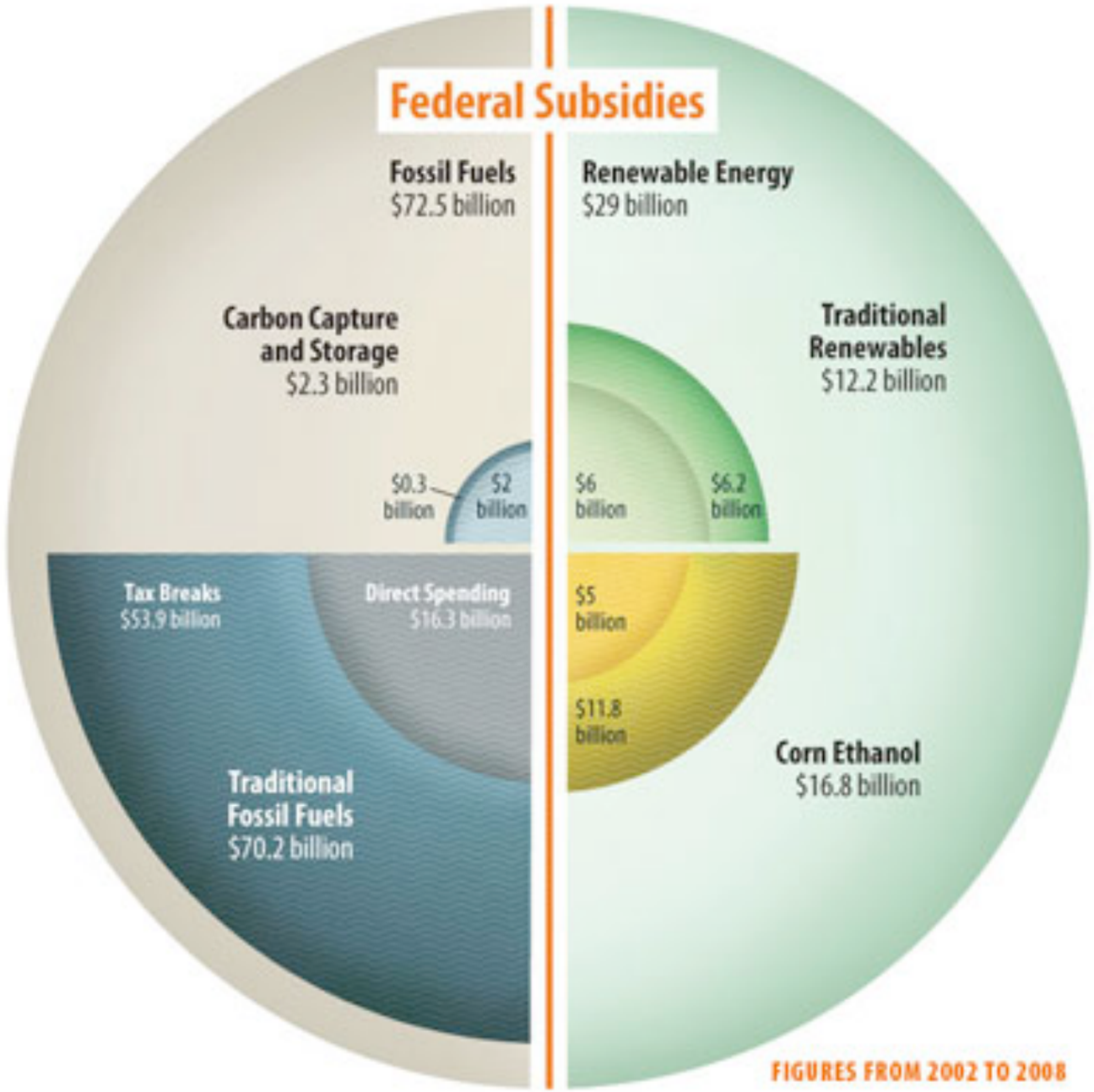
Judge relative size of circles in relation to others?

Judge relative distances between circles?



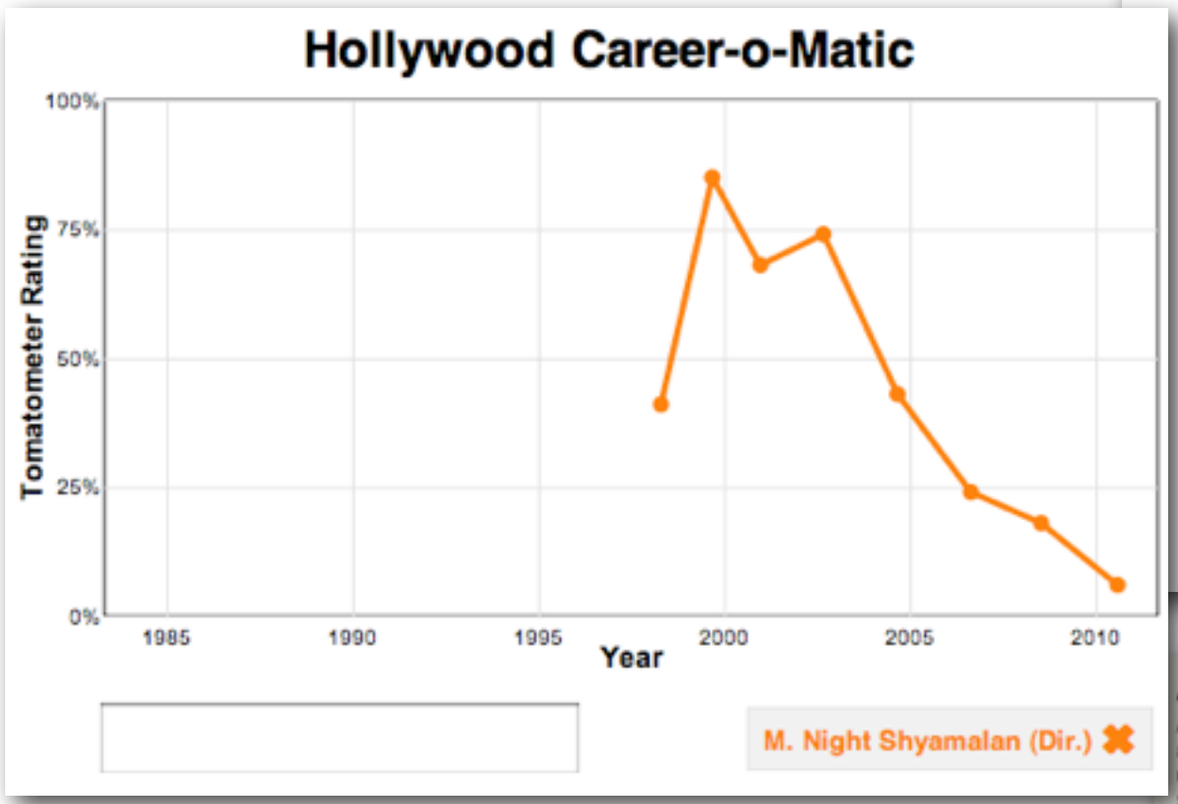
Judge relative sizes of circles within circles?

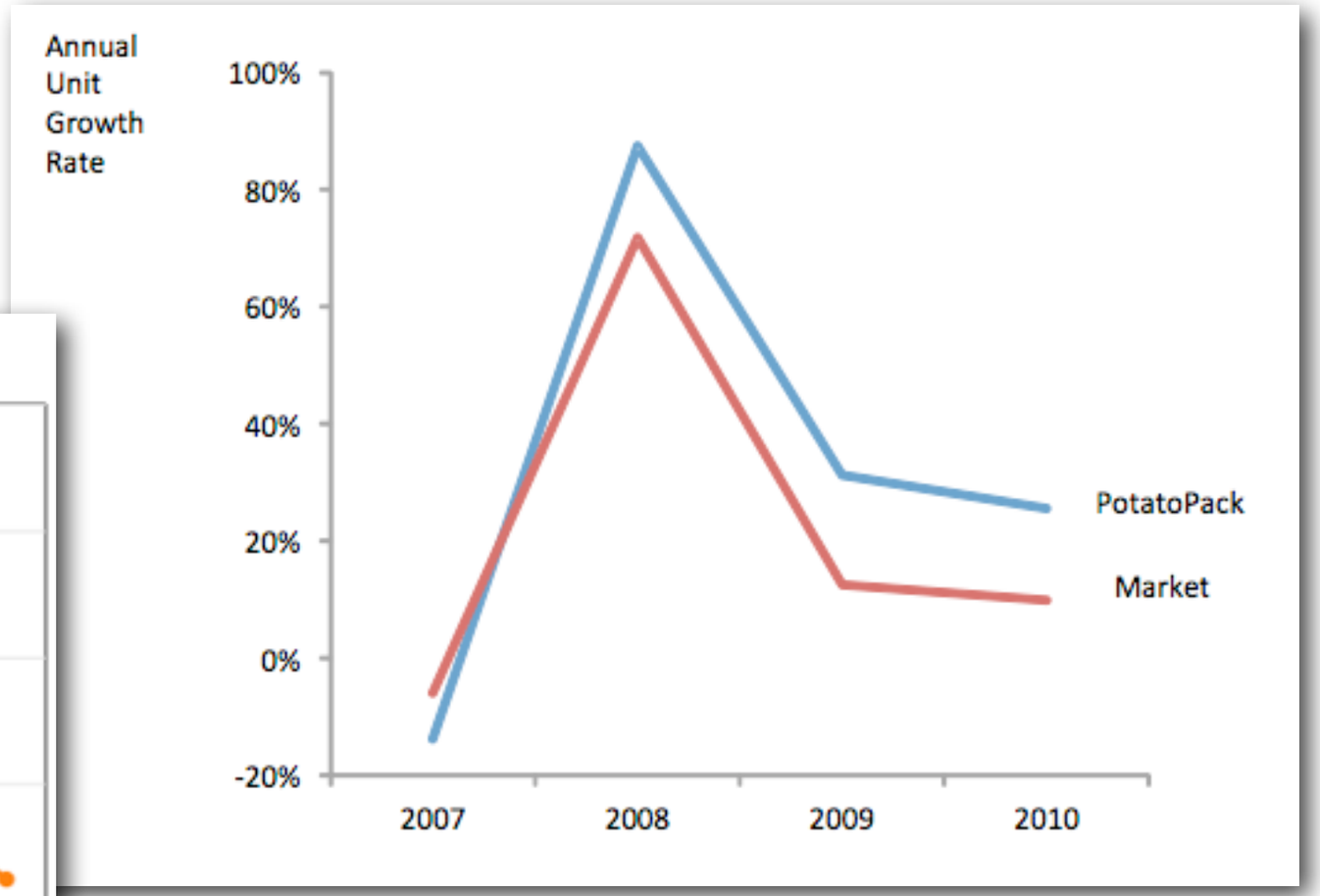




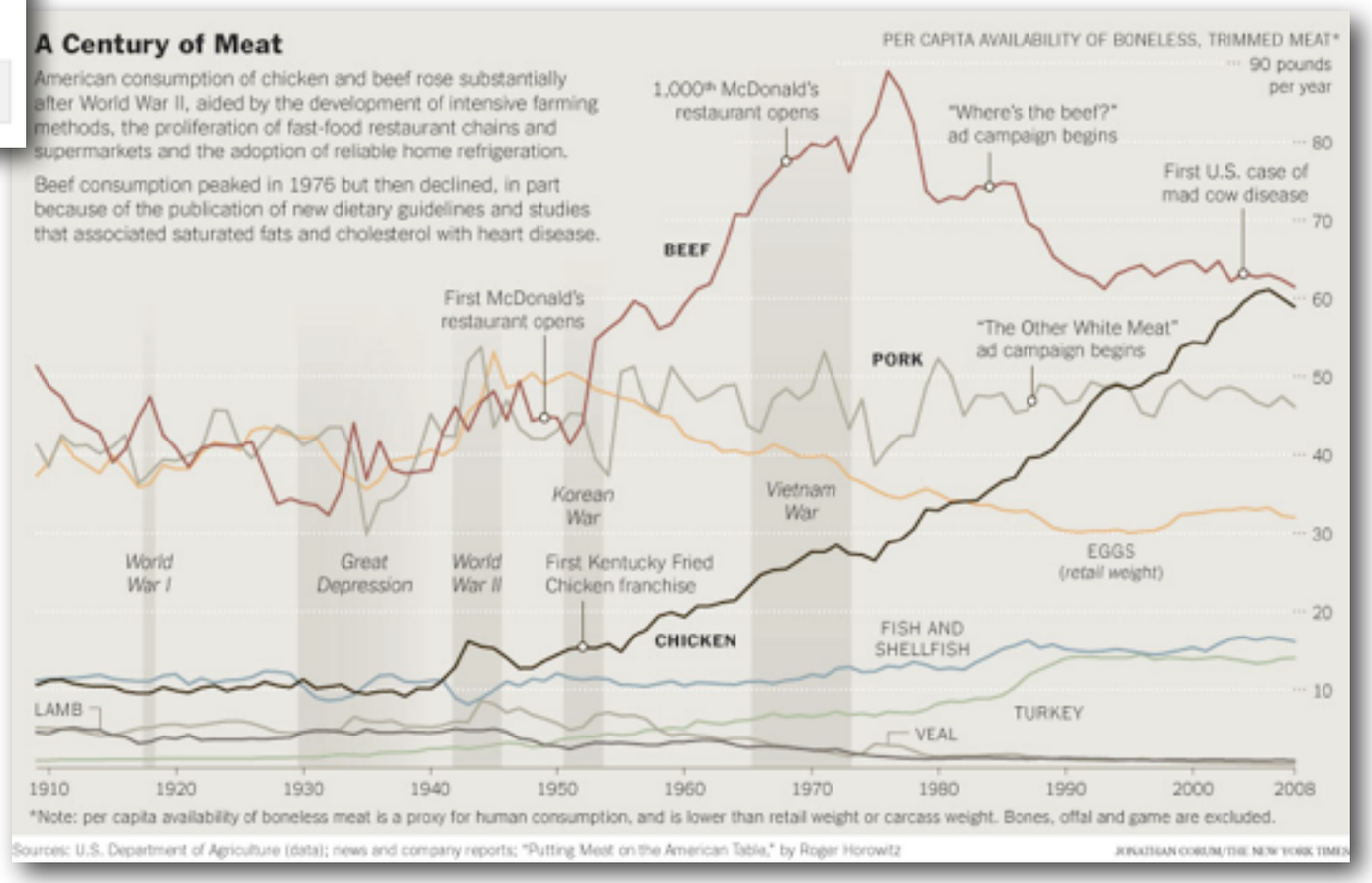
Peter and Maria Hoey (Source: Tommy McCall/Environmental Law Institute)

line graphs

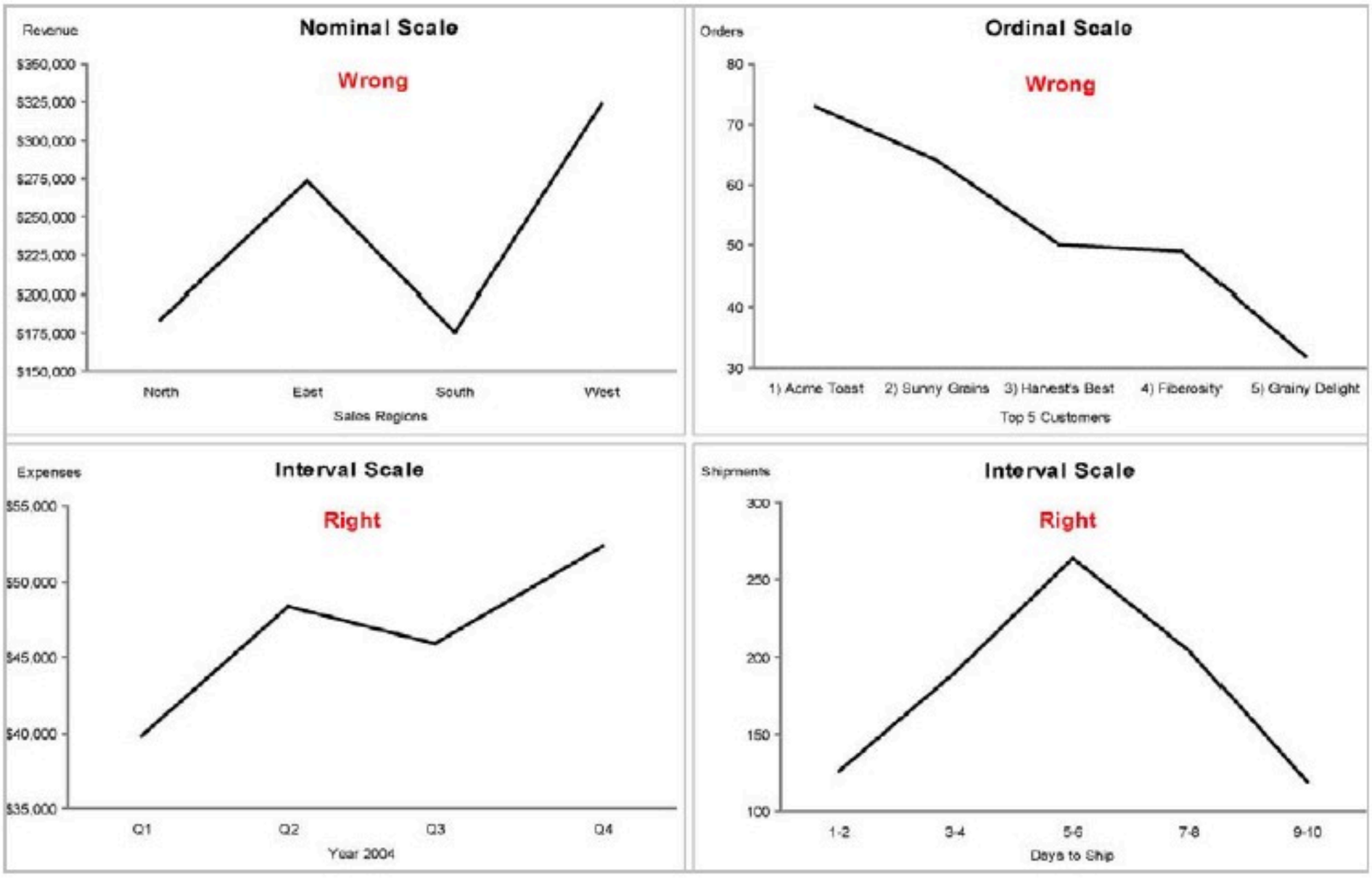


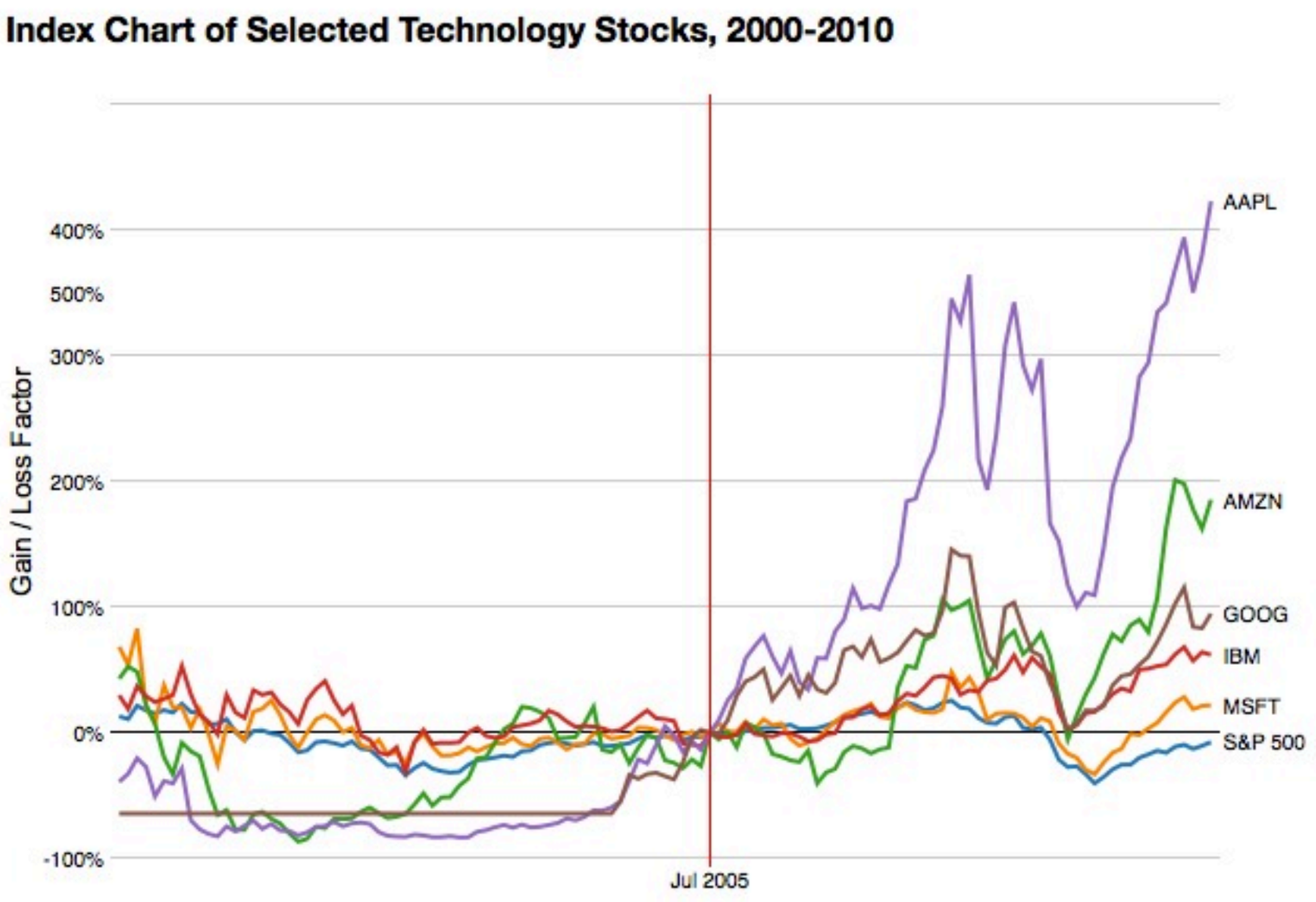


http://flowingdata.com/2011/06/14/rottentomatoes-trends-with-career-o-matic/



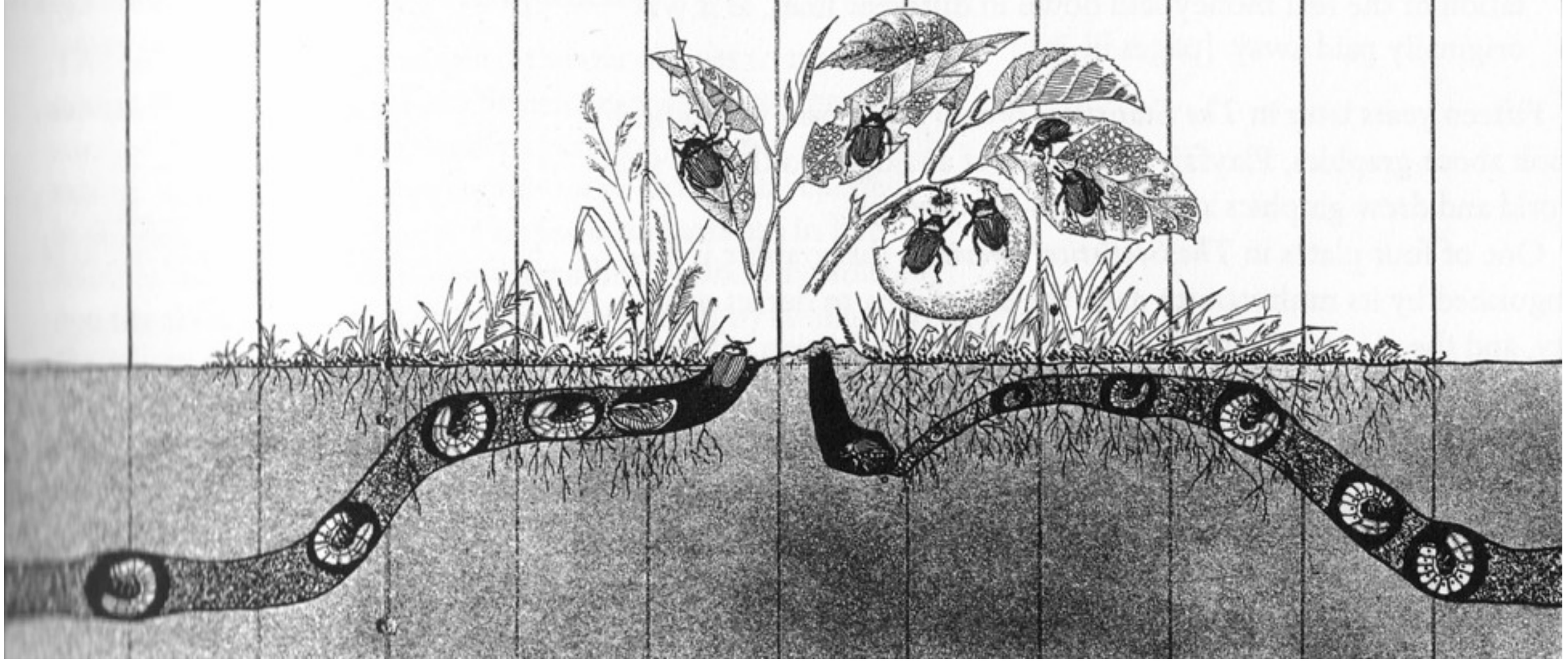
Categorical Data vs. Quantitative Data In Line Graphs



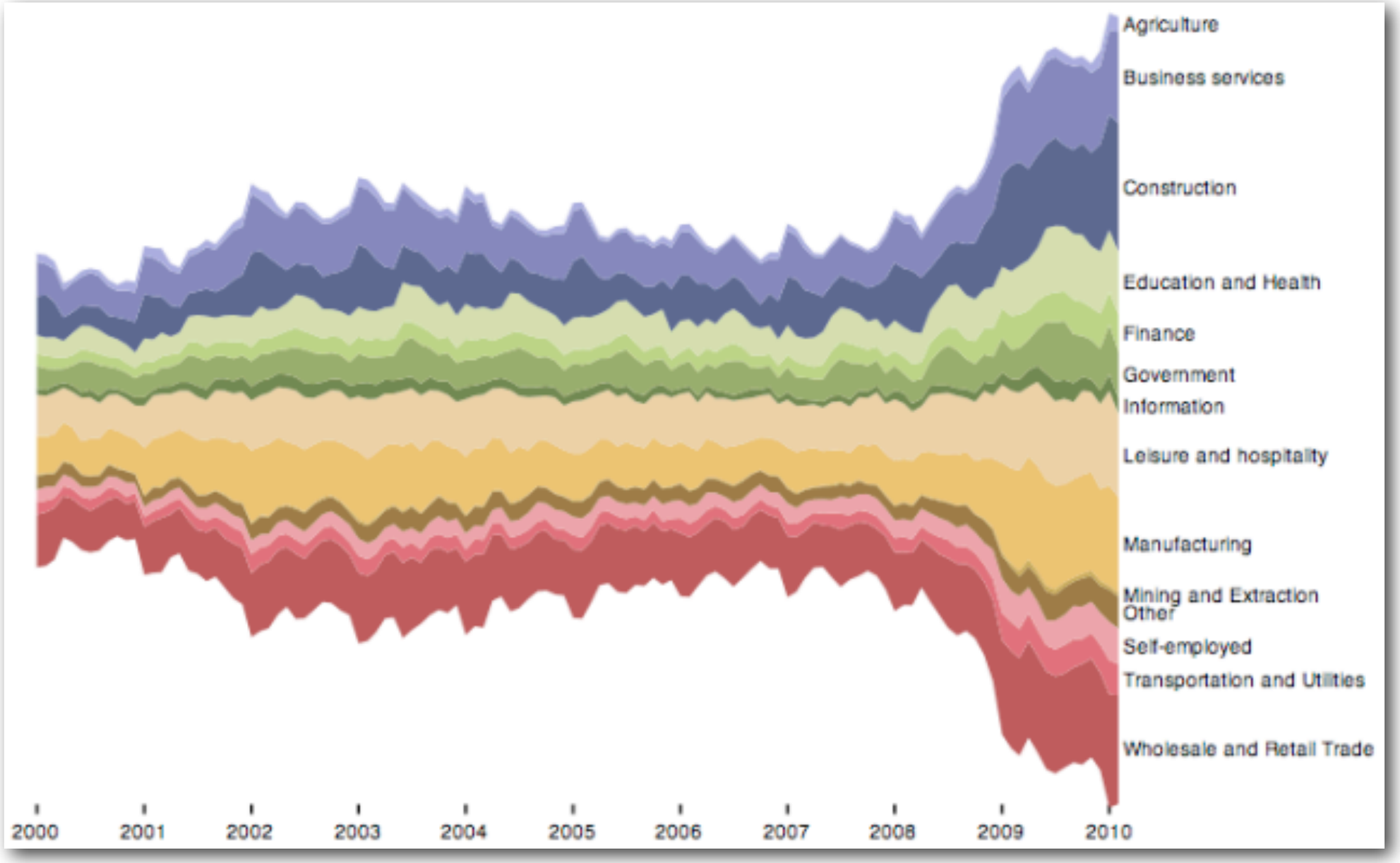


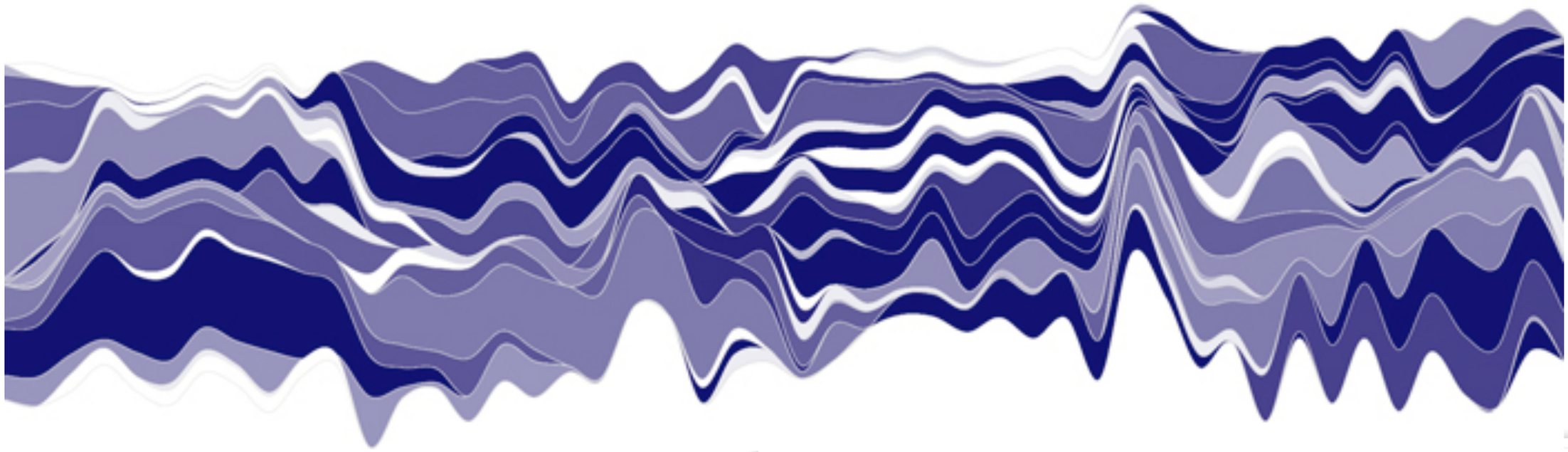
http://hci.stanford.edu/jheer/files/zoo/ex/time/index-chart.html

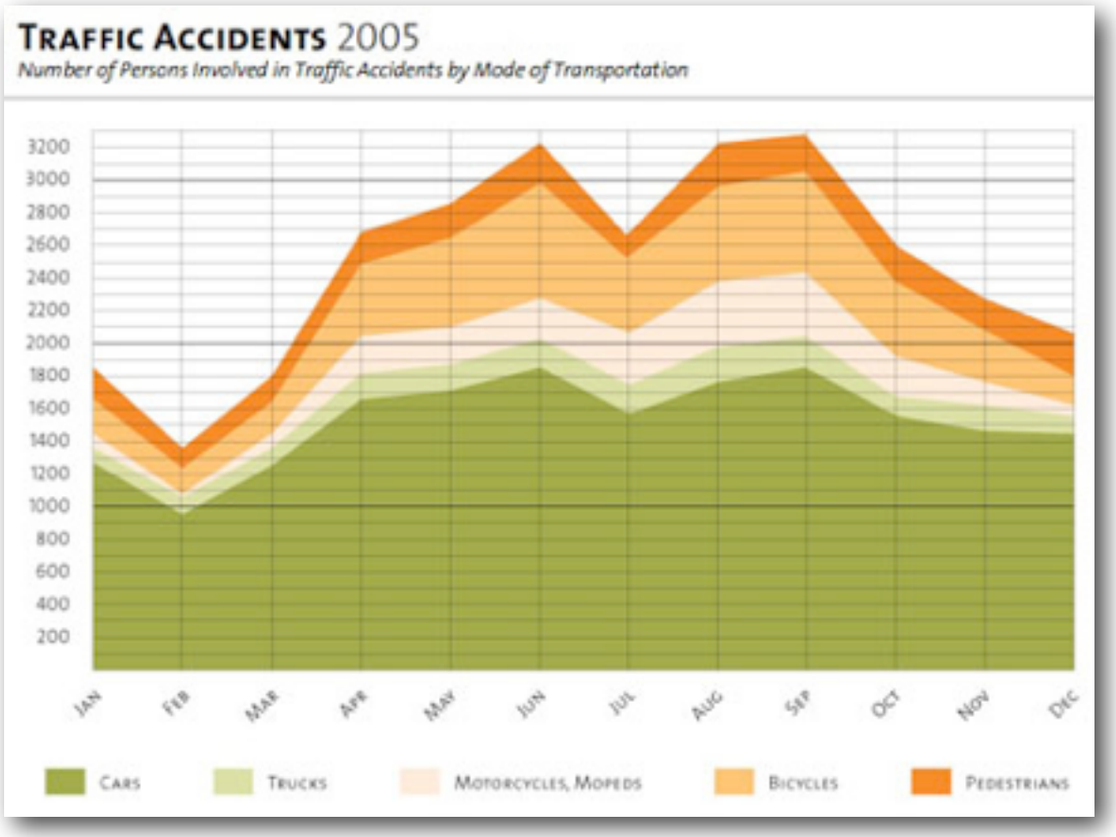
January February March April May June July August September October November December

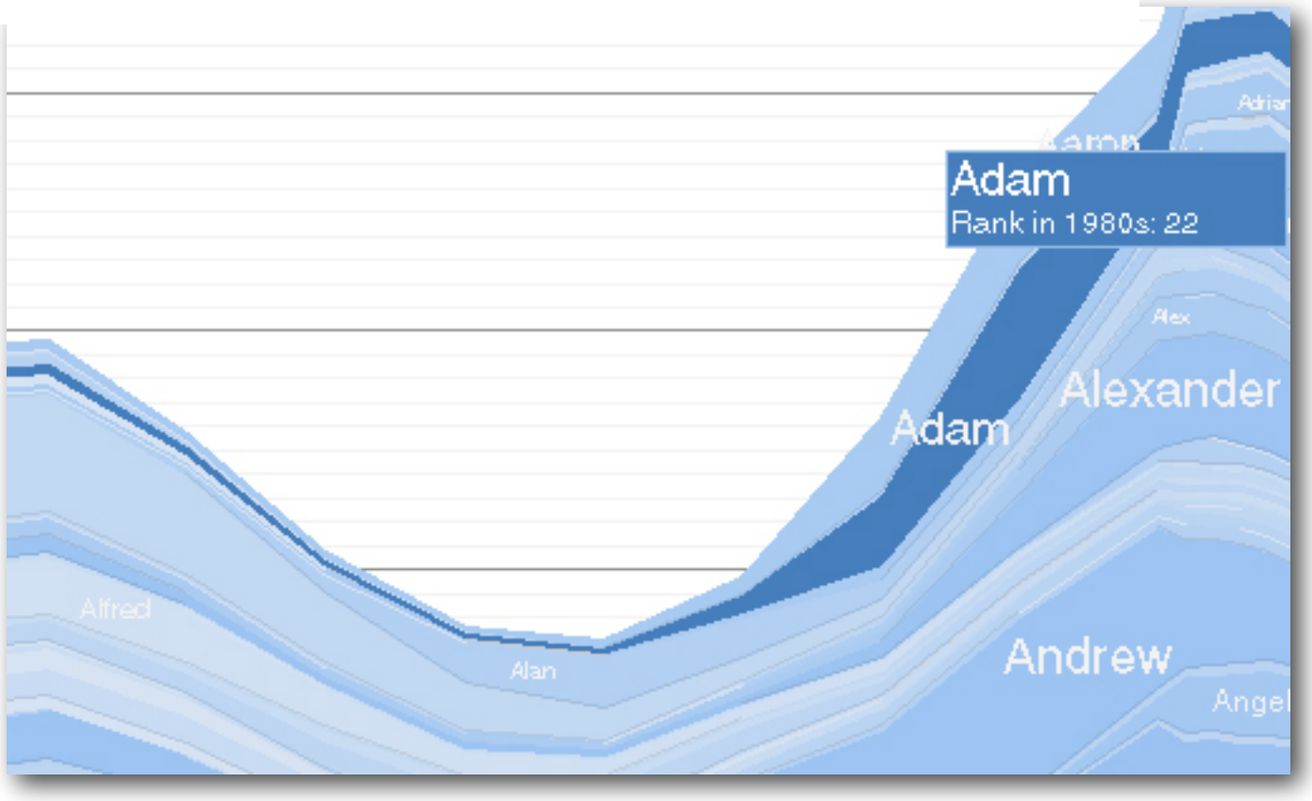


stacked graph







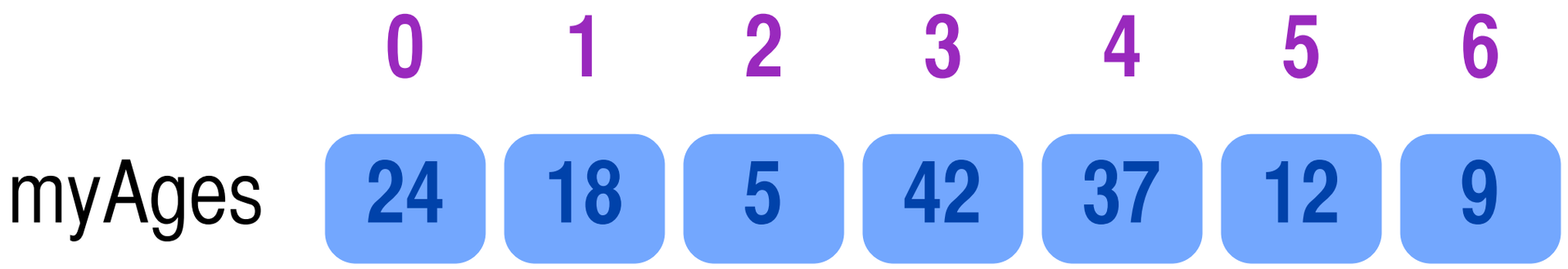


demo: baby name voyager

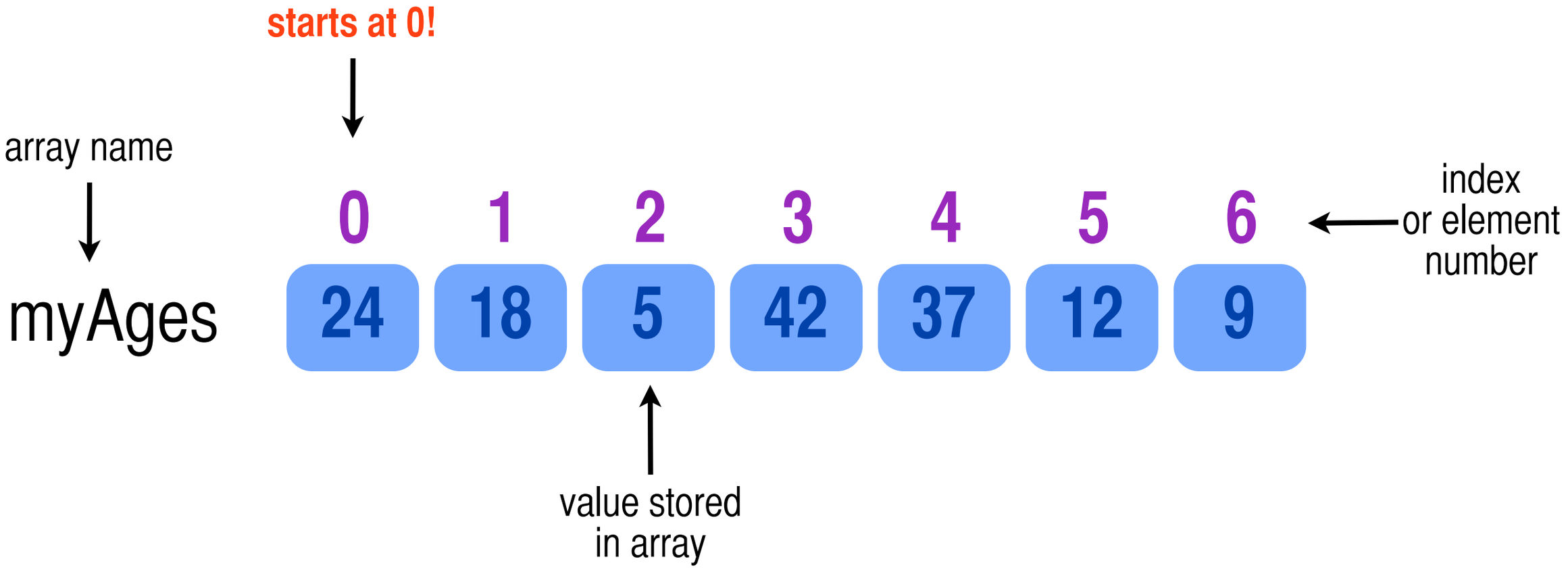
Processing! Arrays.

Arrays in Processing!

- Our first data structure! We'll store **data** there.
- An array is an **ordered, indexed** list of items
- Arrays allow us to use lots of variables without having to keep track of the names of each variable
- You can think of an array as a bank of lockers or a row of compartments in which to store information..

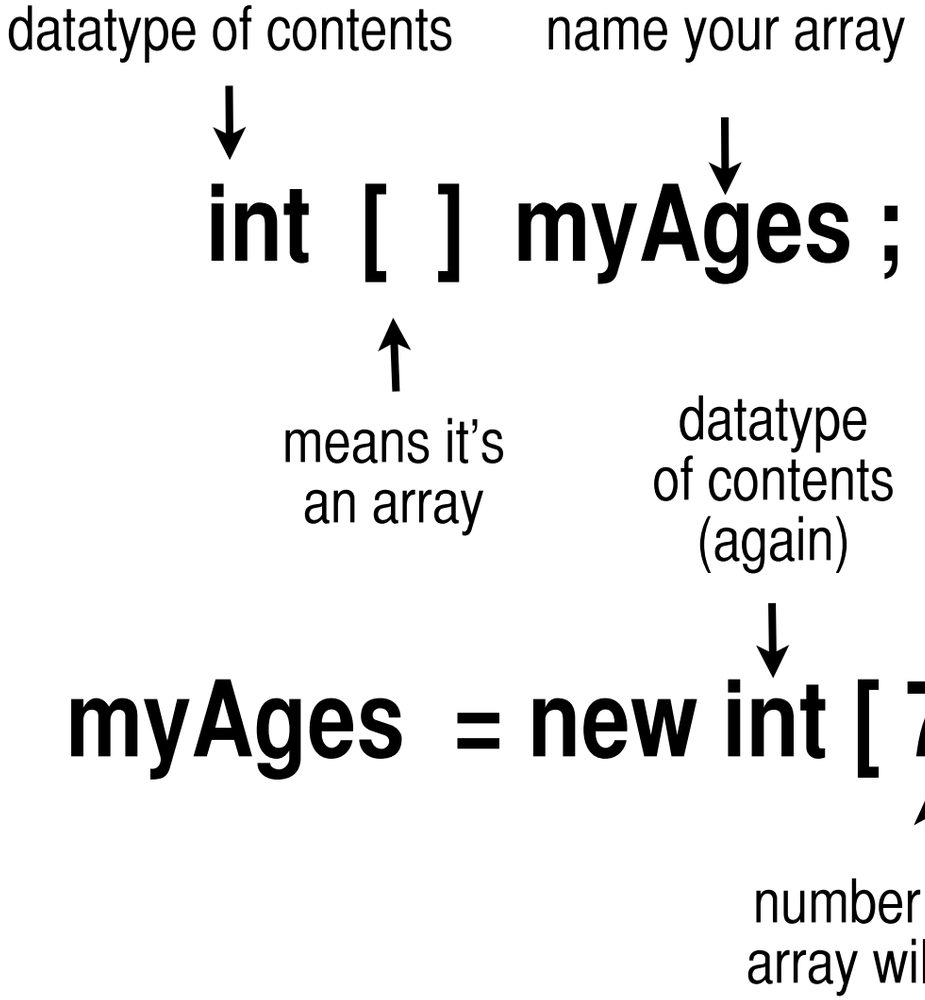


Array Anatomy

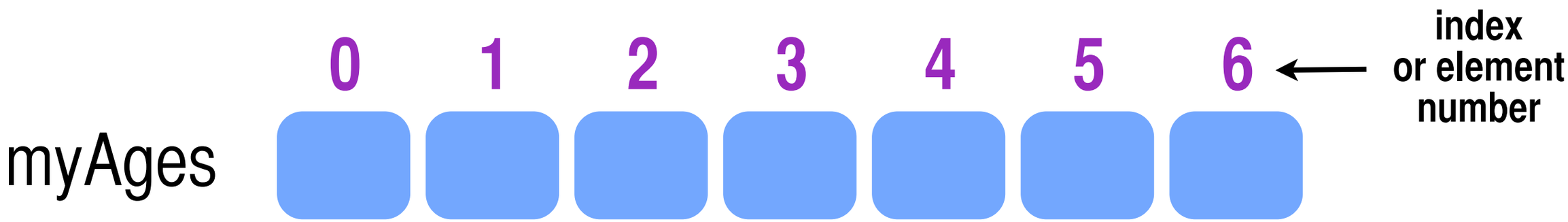


Declare and Initialize Your Array

1. declare:

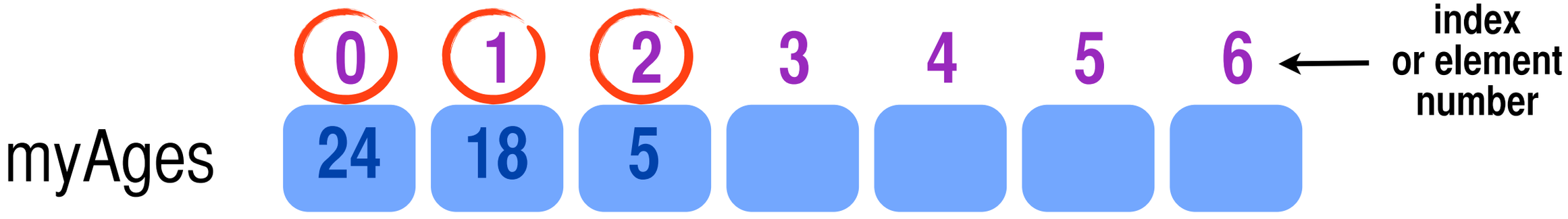


2. initialize:



Putting stuff in your array
… use the **index**

```
myAges[0] = 24;
myAges[1] =18;
myAges[2] =5;
...
```



Or, all at once...

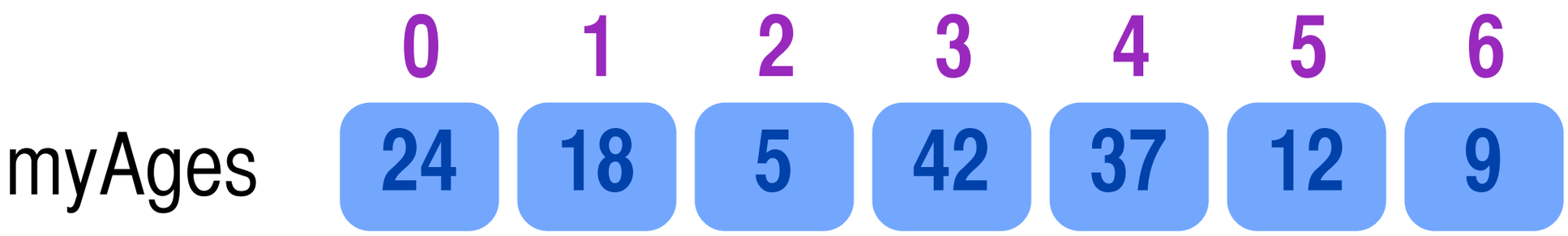
datatype of contents

name your array

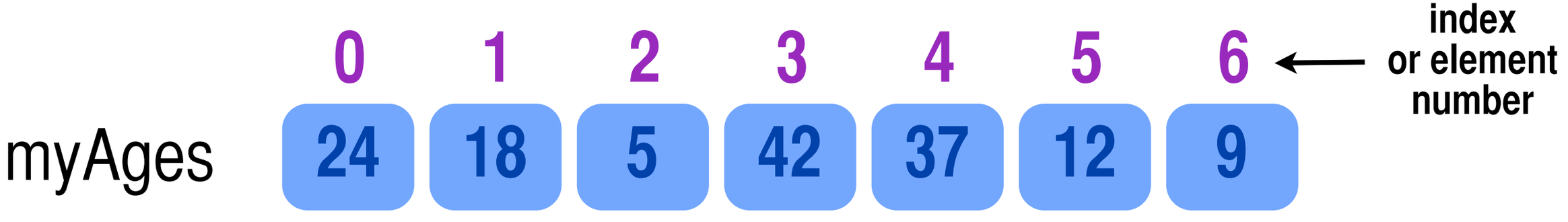
```
int [ ] myAges = { 24, 18, 5, 42, 37, 12, 9}
```

our data inside!

means it's an array

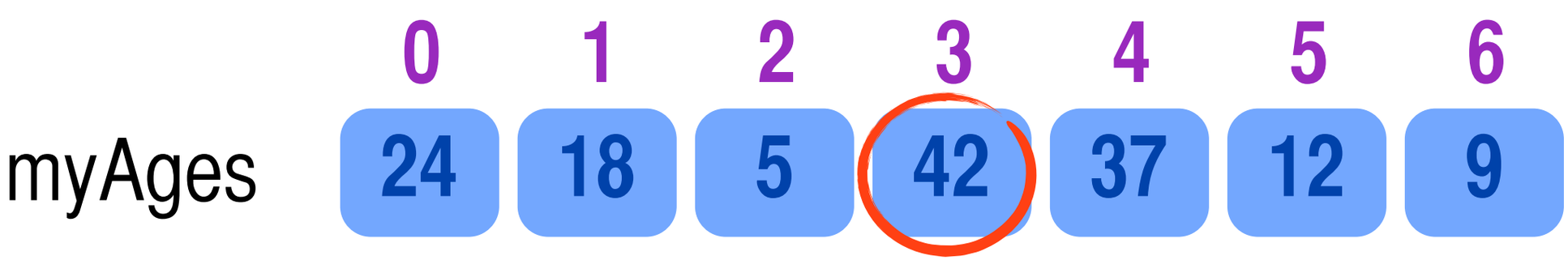


Getting stuff out: use your **index**



use this structure: arrayName[index]

myAges[3] will give us **42**, which is stored in the 4th compartment of the array



A new kind of variable

- We'll use the format **myArray[index]** as a new kind of variable
- We'll use this notation as a stand-in for a variable name
- We have access to every item of data in our array using this format

The **For** loop!

Use a for loop to do something a certain number of times.
We add one to the value of i each time through the loop.
This value is GREAT for accessing items in an array.

test condition

start condition

increment
(add one to i)

```
for (int i = 0; i <10; i++)
{
    do something;
}
```

← this gets done 10 times!

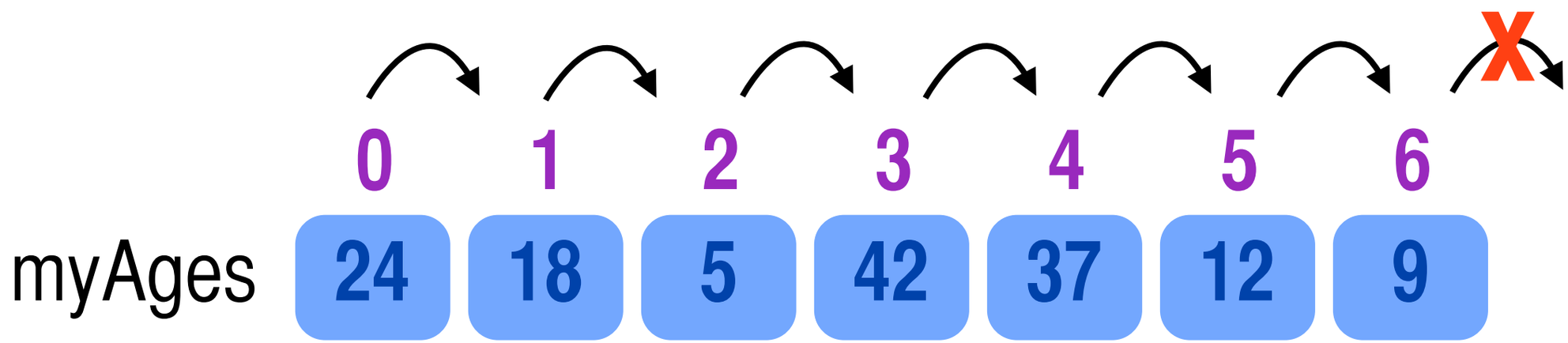
Access your array in a loop- *but don't go out of bounds!*

the array knows its own length.
it's a variable, accessed like this: **arrayName.length**

use the length of the array as the test condition for the loop

```
for (int i= 0; i<myAges.length; i++)
 {
    println(myAges[i]);
 }
```

← just print out each value stored in the array



Examples

- **simpleArray.pde** : declare, intialize and load the array all in one step. print out the values and the current index of the array to the console window using println( ).
- **classAgesArray.pde** : Use the values in the array to draw ellipses on the screen.
- **classAgesLoadStrings.pde:** Use the loadStrings() function to load external data into the array.

loadStrings()

- function that brings data from an external .txt file and **returns an array of strings**
- add your .txt file to your sketch folder Sketch>Add File
- loadStrings( ) will put **one line** from the text file into each position in the array
- If your data is numerical, you need to **convert it from strings to ints** before you use it.

loading Strings

datatype of contents

name of the new array

name of text file in quotes must be in sketch data folder

```
String [ ] myStringData = loadStrings(“ages.txt” );
```

call to load strings

ParseInt( )

- The parseInt( ) function takes a number that is in string form “45” “2333” and converts it into a functional integer

new variable for int

string

```
int newInt = parseInt(“77”);
```

using string array to get the data

```
newInt = parseInt(myStringData[0]);
```

Homework

- Look at the three example sketches for this week, read through the comments and code until you have a good understanding of how it works
- Email me if you have any questions
- See if you can modify myAgesArray.pde to include the names of your classmates below the circle representing their ages. **Make a string array and use the text() function in the loop to print the names**
- Extend the functionality of the above program to load an external text file with your classmate's names. You'll have to create the text file, add it to the sketch, and use loadStrings( ) to get it into an array.
- Reading on Arrays, if needed: **Ch. 10** in *Getting Started With Processing* (pp141–150); **Ch. 9** in *Learning Processing* (pp141–162)